This CVPR workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

Scene-Text-Detection Method Robust against Orientation and Discontiguous Components of Characters

Rei Endo, Yoshihiko Kawai, Hideki Sumiyoshi, Masanori Sano
Science and Technology Research Laboratories, NHK, Tokyo, Japan
{endou.r-mm,kawai.y-lk,sumiyoshi.h-di,sano.m-fo}@nhk.or.jp

Abstract

Scene-text detection in natural-scene images is an important technique because scene texts contain location information such as names of places and buildings, but many difficulties still remain regarding practical use. In this paper, we tackle two problems of scene-text detection. The first is the discontiguous component problem in specific languages that contain characters consisting of discontiguous components. The second is the multi-orientation problem in all languages. To solve these two problems, we propose a connected-component-based scene-text-detection method. Our proposed method involves our novel neighbor-character search method using a synthesizable descriptor for the discontiguous-component problems and our novel region descriptor called the rotated bounding box descriptors (RBBs) for rotated characters. We also evaluated our proposed scene-text-detection method by using the well-known MSRA-TD500 dataset that includes rotated characters with discontiguous components.

1. Introduction

Text in images contains valuable semantic information for various content-based applications [1, 6, 24]. A representative one is location information, and texts of images taken in town often contain clues to identify location. Traffic signs are located in many places in the city, and they include names of streets and places. Buildings and shops have put up signboards with their names at the entrance. By recognizing such texts in images, we can use information of the texts to identify locations. These location information can be used for city navigation and autonomous driving. In addition, since information can be obtained only from image data, it can be used in indoor or underground where GPS can not be used. Therefore, a text recognition method is useful for location identification from images.

Text-recognition methods have been extensively investigated and researchers have proposed effective ones. In particular, an optical-character-recognition method for recognizing printed text in scanned documents is one of the most successful [11]. However, it still difficult for computers to detect and recognize *scene text*, which is text in natural-scene images because scene text has various interference factors such as appearance variation (*e.g.* changes in character size and font), language, orientation, distortion, noise, occlusion, and complex background [11]. In this paper, we tackle two problems caused by interference factors in scene-text detection. One is the discontiguous-component problem and the other is the multi-orientation problem.

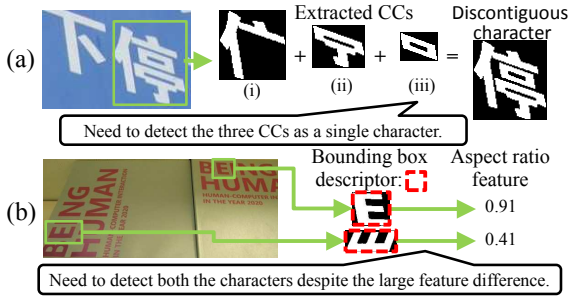

Figure 1. (a) Discontiguous-component and (b) multi-orientation problems. These problems confuse scene-text-detection methods using geometrical features.

The discontiguous-component problem is caused by characters with discontiguous components (hereafter called discontiguous characters). A discontiguous character consists of multiple connected components (CCs), as shown in Fig. 1. Geometrical features, which are important clues for text detection (*e.g.* size and aspect), calculated from a single CC of a discontiguous character, greatly differ from that of a contiguous character, and the difference easily leads to misdetection. While "i" and "j" are the only discontiguous characters in English, Chinese and Japanese have a large number of discontiguous characters. The multi-orientation problem is caused by various text orientations of scene text. Most general geometrical features of characters are not invariant to the orientation of the characters, as shown in Fig. 1. Therefore, it is necessary to solve the two problems for detecting rotated scene text written in languages such as Chinese and Japanese. Note that *text* means a word or sentence consisting of more than one character.

Many scene-text-detection methods have been proposed,

but most target horizontal English texts. Several methods detect texts, including discontiguous characters, based on the uniformity of text alignment and size [16]. However, if they are used to detect rotated text, including discontiguous characters, a large number of false detections will occur. With the existing methods based on geometrical features affected by rotation, it is difficult to correctly eliminate false detections. However, other methods estimate the text orientation [25, 28]. These methods also use the uniformity of text alignment so that when there are texts containing a discontiguous character, orientation estimation is difficult due to the uncertainty of whether a small region, *i.e*., discontiguous component, forms a character, a part of a character, or just noise. Furthermore, we found no method for solving the two problems at the same time in our survey.

In this paper, we propose a CC-based scene-text-detection method robust against rotated discontiguous characters. Our method uses our novel neighbor-character search method with synthesizable descriptors to solve the discontiguous-component problem. The synthesizable descriptor is a particular class of descriptors that can be combined using the simple operation which lower calculation cost. Our method calculates a descriptor of a discontiguous character from only those of the individual CCs constituting the character by using synthesizable descriptors. To solve the multi-orientation problem, our method uses our novel region descriptors called the rotated bounding box descriptors (RBBs), which are synthesizable descriptors. RBBs represent coordinates of bounding boxes around regions rotated through multiples of a fixed angles, and are used to accurately calculate geometrical features of the regions. By using our neighbor-character search method with RBBs and other synthesizable descriptors, our method can detect rotated scene text that includes discontiguous characters.

The rest of paper is organized as follows. We give a brief review on related work on CC-based methods in Sec. 2 and describe our proposed scene-text-detection method in Sec. 3. We explain the experimental results and provide discussion in Sec. 4 followed by conclusions in Sec. 5.

2. Related Work

Existing scene-text-detection methods are roughly classified into two types: sliding-window based [3, 9, 10, 19] and CC based [2, 5, 11, 13, 15, 16, 21, 22, 23, 25, 26, 27]. Sliding-window-based methods use a sliding window to search for character or text regions in an image then use machine-learning techniques to identify text. Although these methods can generally detect discontiguous characters, it is difficult for them to handle large variations in character size and aspect. Because sliding-windows detect only characters the same size and aspect as those included in the training data. CC-based methods extract CCs as character candidates and form groups of character candidates with similar features. These methods can handle the variation, but existing CC-based methods do not take into account of the discontiguous component problem. We focus on CC-based methods to handle various characters and describe related work. Maximally stable extremal region (MSER)-based methods have been particularly successful among CC-based methods.

A large number of MSER-based methods perform well in scene-text detection [5, 15, 16, 25, 27]. Because they can extract extremal regions, which are a kind of CC, that are robust against size variation, low resolution, and complex background[12]. Neumann and Matas used a multi-stage character classifier to efficiently reduce the number of false character candidates [15]. Yin *et al*. reduced the number of CCs until the overlapping region was eliminated by using the aspect ratio and stability of CCs [27]. However, their methods do not take into account the presence of discontiguous characters. Unlike English, languages such as Chinese and Japanese contain many discontiguous characters. In some cases, continuous characters can unexpectedly divided into multiple CCs due to the limitation of CC extraction algorithms. Neumann and Matas's method can detect such characters using an estimation of a text's base line [16]. However, it is difficult to estimate a text's base line when the text is rotated.

Recently, deep neural network (DNN)-based methods, which are hybrid methods of sliding-window-based and CC-based methods, also have been successfully applied in various computer vision tasks [1, 7, 18, 28, 29]. Although DNN-based methods require a huge amount of training data and high computational performance, they can detect characters from various images with high accuracy. Bissacco *et al*. developed a system using a deep-learning method with histogram of gradient features, and the system can detect characters in uncontrolled situations [1]. He *et al*. proposed a text-attentional convolution neural network that can learn a text region mask, character label, and binary text/non-text information in uniform schema [7]. These methods work well in English but are still affected by the discontiguous component problem. One reason is that some languages that include discontiguous characters have many types of characters. Large amounts of training data and computational resources are required to correctly classify these characters. GPU can solve the problem about computational resources, but we are also assuming use with devices that cannot use GPU such as smart-phones. Another reason is that the shapes of some CCs constituting discontiguous characters are often the same as those of contiguous characters (*e.g*. the CC (iii) in Fig. 1 is similar to "O" or zero in English.). There are difficulties in distinguishing them even when a DNN is used.

The multi-orientation problem still needs to be solved. Most existing methods use geometrical features to detect text candidates from detected characters based on the assumption that text orientation is horizontal or near-horizontal. Yin *et al*. proposed an adaptive clustering method that uses the geometrical and orientation features of

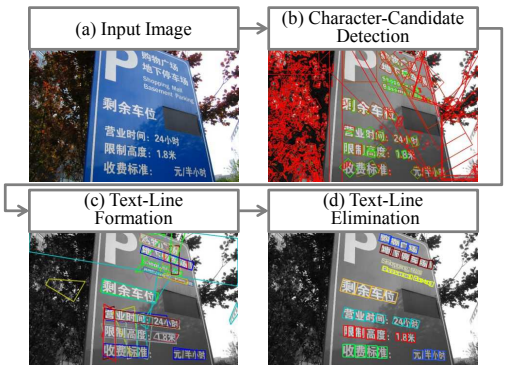

Figure 2. Overview of proposed method and sample results of each processing stage. (a) Input image. (b) Green rectangles represent CCs with high character probability, and red ones represent those with low character probability. (c) Randomly colored rectangles represent text-line candidates. (d) Rectangles represent each single character of detected text lines. Characters with same color rectangles belong to same detected text line.

character candidate pairs to detect rotated texts [25]. Zhang *et al.*'s method detects rotated texts by using geometrical features and information of a text-block, which is detected by text-block fully constitutional networks [28]. However, the geometrical features of existing methods are not accurate because they use a non-rotated bounding box or a circumscribed circle. When an elongated shaped character is oriented, the non-rotated bounding box changes greatly, as shown in Fig. 1. The radius of the circumscribed circle is used as the geometrical distance of a character pair, but the circumscribed circle is affected by character shape. For example, in detecting the text of "Alice," existing methods evaluate the distance values of "Al," "li," "ic", and "ce" under the assumption that the characters belong to the same text when their distance values are similar. However, a circumscribed circle's size of "Al" is much larger than that of "li"; therefore, the detection of "Alice" fails.

3. Proposed Scene-Text-Detection Method

Figure 2 gives an overview of our method. At the beginning of all processing, CCs are extracted from an input image and classified as characters or non-characters. Then, text-line candidates are detected using our neighbor-character search method with RBB and other synthesizable descriptors. Finally, overlapping text-line candidates are throttled into one candidate based on text probabilities. Since the key ideas regarding our method are synthesizable descriptors and RBB, which contribute to our neighbor-character search method, we describe them first then the details of each processing stage. Note that descriptors and features are clearly distinguished in this paper. A descriptor is a storing format for characteristics of a region. A feature is a value or a set of values actually used by the classifiers. For example, a bounding box is a descriptor, and the aspect ratio calculated by the bounding box is a feature.

3.1. Region Descriptors and Features

3.1.1 Synthesizable Descriptors

We propose synthesizable descriptors for efficient computation of compound region descriptors, inspired by the incrementally computable descriptor [15]. The compound region is a region consisting of multiple CCs, where a region of a discontiguous character is always a compound region. By using a synthesizable descriptor, descriptors of a compound region consisting of CCs can be simply calculated from only descriptors of the CCs. Let $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ be regions consisting of one or more CCs, $\boldsymbol{r}_c$ be the compound region of $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$, and $\psi(\boldsymbol{r})$ be a descriptor calculation function of $\boldsymbol{r}$. When the descriptor is a synthesizable descriptor, there exists $\oplus$ satisfying $\psi(\boldsymbol{r}_c) = \psi(\boldsymbol{r}_i) \oplus \psi(\boldsymbol{r}_j)$, where $\oplus$ denotes an operation that combines the descriptors of $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$. The proposed scene-text-detection method calculates descriptors of compound regions using a synthesizable descriptor to reduce calculation cost.

The following synthesizable descriptors are used with our proposed method for compound regions: RBB and horizontal continuity descriptors and other typical descriptors transformed into synthesizable descriptors. The $\oplus$ of a descriptor that is not specifically mentioned is an addition $(+)$.

- **RBB** $(y^{\min}_{\theta_m}(\boldsymbol{r}), y^{\max}_{\theta_m}(\boldsymbol{r}), x^{\min}_{\theta_m}(\boldsymbol{r}), x^{\max}_{\theta_m}(\boldsymbol{r}))$. RBB represents coordinates of a bounding box rotated to an arbitrary angle θ_m as shown by Fig. 3. The RBB is used to accurately calculate the geographical features of a region rotated to θ_m. Each value is defined as

$$y^{\min}_{\theta_m}(\boldsymbol{r}) = \min_{r\in\boldsymbol{r}}\{\tan(\theta_m)x_r + y_r\}, \quad (1)$$

$$y^{\max}_{\theta_m}(\boldsymbol{r}) = \max_{r\in\boldsymbol{r}}\{\tan(\theta_m)x_r + y_r\}, \quad (2)$$

$$x^{\min}_{\theta_m}(\boldsymbol{r}) = \min_{r\in\boldsymbol{r}}\{x_r + \tan(\theta_m)(h - y_r)\}, \quad (3)$$

$$x^{\max}_{\theta_m}(\boldsymbol{r}) = \max_{r\in\boldsymbol{r}}\{x_r + \tan(\theta_m)(h - y_r)\}, \quad (4)$$

  where x_r and y_r are coordinates of a pixel r in a region $\boldsymbol{r}$, h is the height of the input image, and θ_m is obtained from a number of calculation angles, $\mathrm{N}^{\mathrm{ori}}$, which is a predefined fixed value. The θ_m is defined as

$$\theta_m = \frac{\pi}{2} \times \frac{m}{\mathrm{N}^{\mathrm{ori}}} \quad (m = \{0, 1, ..., \mathrm{N}^{\mathrm{ori}}\}). \quad (5)$$

  We empirically set $\mathrm{N}^{\mathrm{ori}} = 6$. The $\oplus$ of each RBB values are $\min$, $\max$, $\min$ and $\max$ respectively.
- **Horizontal Continuity** $(\boldsymbol{v}^{\min}_{\boldsymbol{r},\theta_m}, \boldsymbol{v}^{\max}_{\boldsymbol{r},\theta_m})$. Two vectors represent the sections in which the region of $\boldsymbol{r}$ is continuous in θ_m as shown by Fig. 4. The initial vectors of $\boldsymbol{v}^{\min}_{\boldsymbol{r},\theta_m}$ and $\boldsymbol{v}^{\max}_{\boldsymbol{r},\theta_m}$ are $\{x^{\min}_{\theta_m}(\boldsymbol{r})\}$ and $\{x^{\max}_{\theta_m}(\boldsymbol{r})\}$, respectively. The $\oplus$ is shown in Algorithm 1.
- **Area** $n^{ara}(\boldsymbol{r})$[15]. The number of pixels.
- **Total Color** $(s^{\mathrm{c0}}(\boldsymbol{r}), s^{\mathrm{c1}}(\boldsymbol{r}), s^{\mathrm{c2}}(\boldsymbol{r}))$. The totals of pixel values in HSI color space.
- **Number of Stroke-Support Pixels (SSPs)** $n^{\mathrm{ssp}}(\boldsymbol{r})$. An SSP [16] is used to calculate the stroke width.

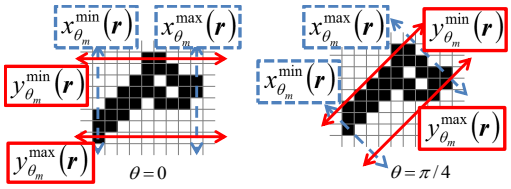

Figure 3. Rotated-bounding boxes and RBBs when $\theta = 0$ and $\pi/4$

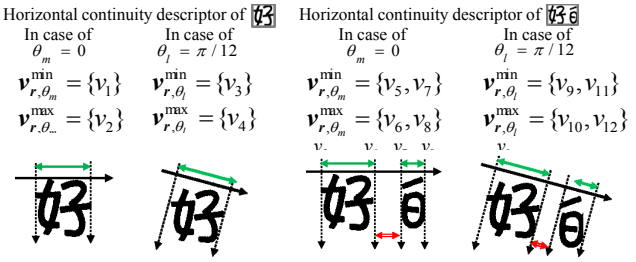

Figure 4. Examples of horizontal continuity descriptor. Green lines represent horizontal continuity descriptors, and red ones represent discontinuity features calculated by the descriptors.

- **Total Stroke Width** $s^{\mathrm{swd}}(\boldsymbol{r})$. The total of stroke-width value of SSP.
- **Perimeter** $n^{\mathrm{peri}}(\boldsymbol{r})$[15]. The length of the boundary.
- **Total Smoothness Value** $s^{\mathrm{smt}}(\boldsymbol{r})$. The total of difference values of adjacent boundary pixels' gradient direction. The details of smoothness features are given in previous studies [27].

For RBB, Eqs. 1 and 2 represent the min and max values of the vertical position, and Eqs. 3 and 4 represent the min and max values of the horizontal position in the θ_m-rotated coordinate system, as shown in Fig. 3. Therefore, RBB of θ_m allows for accurate calculation of $\boldsymbol{r}$'s size, aspect, and position in the θ_m-rotated coordinate system. As $\mathrm{N}^{\mathrm{ori}}$ increases, geometrical features with more various angles can be calculated, but the total RBB calculation cost of all CCs increases. However, RBB is incrementally computable, which is a particular class of descriptors that can be quickly calculated for all CCs extracted by MSER algorithm[15]. The descriptor's computing complexity of all CCs in an image of N pixels is $O(N)$. In our experiments, total RBB computation time of all CCs with $\mathrm{N}^{\mathrm{ori}} = 6$ was only 0.07 seconds longer than the time with $\mathrm{N}^{\mathrm{ori}} = 1$.

3.1.2 Features from Synthesizable Descriptors

Various geometrical features in the θ_m-rotated coordinate system can be obtained from RBB of θ_m. The basic features for a single region and region pair are as follows.

Some size features of $\boldsymbol{r}$ can be simply calculated from RBB of a region $\boldsymbol{r}$. The rotated height $h_{\theta_m}(\boldsymbol{r})$ and rotated width $w_{\theta_m}(\boldsymbol{r})$ are calculated as

Algorithm 1 $\oplus$ of horizontal continuity of $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ constituting $\boldsymbol{r}_c$ for certain θ_m

Input: $\boldsymbol{v}^{\min}_{\boldsymbol{r}_i,\theta_m}, \boldsymbol{v}^{\max}_{\boldsymbol{r}_i,\theta_m}$: Continuity descriptor of $\boldsymbol{r}_i$
Input: $\boldsymbol{v}^{\min}_{\boldsymbol{r}_j,\theta_m}, \boldsymbol{v}^{\max}_{\boldsymbol{r}_j,\theta_m}$: Continuity descriptor of $\boldsymbol{r}_j$
Output: $\boldsymbol{v}^{\min}_{\boldsymbol{r}_c,\theta_m}, \boldsymbol{v}^{\max}_{\boldsymbol{r}_c,\theta_m}$: Continuity descriptor of $\boldsymbol{r}_c$
$\boldsymbol{v}^{\min}_{\boldsymbol{r}_c,\theta_m} \Leftarrow \varnothing$, $\boldsymbol{v}^{\max}_{\boldsymbol{r}_c,\theta_m} \Leftarrow \varnothing$
$q_{\mathrm{i}} \Leftarrow 0$, $q_{\mathrm{j}} \Leftarrow 0$, $v^{\min} \Leftarrow 0$, $v^{\max} \Leftarrow 0$
while $q_{\mathrm{i}} < |\boldsymbol{v}^{\min}_{\boldsymbol{r}_i,\theta_m}|$ or $q_{\mathrm{j}} < |\boldsymbol{v}^{\min}_{\boldsymbol{r}_j,\theta_m}|$ **do**
  if $q_{\mathrm{i}} \geqq |\boldsymbol{v}^{\min}_{\boldsymbol{r}_i,\theta_m}|$ or $\boldsymbol{v}^{\min}_{\boldsymbol{r}_i,\theta_m}[q_{\mathrm{i}}] > \boldsymbol{v}^{\min}_{\boldsymbol{r}_j,\theta_m}[q_{\mathrm{j}}]$ **then**
    $v^{\min} \Leftarrow \boldsymbol{v}^{\min}_{\boldsymbol{r}_j,\theta_m}[q_{\mathrm{j}}]$, $v^{\max} \Leftarrow \boldsymbol{v}^{\max}_{\boldsymbol{r}_j,\theta_m}[q_{\mathrm{j}}]$, $q_{\mathrm{j}} \Leftarrow q_{\mathrm{j}} + 1$
  else
    $v^{\min} \Leftarrow \boldsymbol{v}^{\min}_{\boldsymbol{r}_i,\theta_m}[q_{\mathrm{i}}]$, $v^{\max} \Leftarrow \boldsymbol{v}^{\max}_{\boldsymbol{r}_i,\theta_m}[q_{\mathrm{i}}]$, $q_{\mathrm{i}} \Leftarrow q_{\mathrm{i}} + 1$
  end if
  while **do**
    if $q_{\mathrm{i}} < |\boldsymbol{v}^{\min}_{\boldsymbol{r}_i,\theta_m}|$ and $\boldsymbol{v}^{\min}_{\boldsymbol{r}_i,\theta_m}[q_{\mathrm{i}}] \leq v^{\max} + c_{\theta_m}$ **then**
      $v^{\max} \Leftarrow \boldsymbol{v}^{\max}_{\boldsymbol{r}_i,\theta_m}[q_{\mathrm{i}}]$, $q_{\mathrm{i}} \Leftarrow q_{\mathrm{i}} + 1$
    end if
    if $q_{\mathrm{j}} < |\boldsymbol{v}^{\min}_{\boldsymbol{r}_j,\theta_m}|$ and $\boldsymbol{v}^{\min}_{\boldsymbol{r}_j,\theta_m}[q_{\mathrm{j}}] \leq v^{\max} + c_{\theta_m}$ **then**
      $v^{\max} \Leftarrow \boldsymbol{v}^{\max}_{\boldsymbol{r}_j,\theta_m}[q_{\mathrm{j}}]$, $q_{\mathrm{j}} \Leftarrow q_{\mathrm{j}} + 1$
    end if
    if $v^{\max}$ is not updated **then** break
  end while
  $\boldsymbol{v}^{\min}_{\boldsymbol{r}_c,\theta_m}$ push back $v^{\min}$, $\boldsymbol{v}^{\max}_{\boldsymbol{r}_c,\theta_m}$ push back $v^{\max}$
end while

$$h_{\theta_m}(\boldsymbol{r}) = y^{\max}_{\theta_m}(\boldsymbol{r}) - y^{\min}_{\theta_m}(\boldsymbol{r}) + c_{\theta_m}, \tag{6}$$

$$w_{\theta_m}(\boldsymbol{r}) = x^{\max}_{\theta_m}(\boldsymbol{r}) - x^{\min}_{\theta_m}(\boldsymbol{r}) + c_{\theta_m}, \tag{7}$$

$$c_{\theta_m} = \sqrt{\tan(\theta_m) \times \tan(\theta_m) + 1}, \tag{8}$$

where c_{θ_m} is a coefficient to prevent division by zero and is 1 when the coordinate system is not rotated ($\theta_m = 0$). The length of each coordinate system has a different scale, but a rotated aspect ratio can be calculated without normalization because the lengths of the same coordinate systems can be directly compared.

$$aspect_{\theta_m}(\boldsymbol{r}) = \frac{\min(w_{\theta_m}(\boldsymbol{r}), h_{\theta_m}(\boldsymbol{r}))}{\max(w_{\theta_m}(\boldsymbol{r}), h_{\theta_m}(\boldsymbol{r}))}. \tag{9}$$

RBB of region pair $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ allows accurate calculation of rotated-region pair features: height difference h^{diff}, width difference w^{diff}, horizontal minimum distance x^{dis}, and vertical overlap y^{ovl}. In general, each character size and each interval of character pairs are similar; therefore, these features are useful for rotated text detection. These features of $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ are defined as

$$h^{\mathrm{diff}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\frac{|h_{\theta_m}(\boldsymbol{r}_i)-h_{\theta_m}(\boldsymbol{r}_j)|}{\max(h_{\theta_m}(\boldsymbol{r}_i),h_{\theta_m}(\boldsymbol{r}_j))}, \tag{10}$$

$$w^{\mathrm{diff}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\frac{|(w_{\theta_m}(\boldsymbol{r}_i)-w_{\theta_m}(\boldsymbol{r}_j)|}{\max(w_{\theta_m}(\boldsymbol{r}_i),w_{\theta_m}(\boldsymbol{r}_j))}, \tag{11}$$

$$\begin{aligned}x^{\mathrm{dis}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\max(&x^{\mathrm{min}}_{\theta_m}(\boldsymbol{r}_i)-x^{\mathrm{max}}_{\theta_m}(\boldsymbol{r}_j),\\ &x^{\mathrm{min}}_{\theta_m}(\boldsymbol{r}_j)-x^{\mathrm{max}}_{\theta_m}(\boldsymbol{r}_i))+c_{\theta_m},\end{aligned} \tag{12}$$

$$\begin{aligned}y^{\mathrm{ovl}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\min(&y^{\mathrm{max}}_{\theta_m}(\boldsymbol{r}_i),y^{\mathrm{max}}_{\theta_m}(\boldsymbol{r}_j))-\\ &\max(y^{\mathrm{min}}_{\theta_m}(\boldsymbol{r}_i),y^{\mathrm{min}}_{\theta_m}(\boldsymbol{r}_j))+c_{\theta_m}.\end{aligned} \tag{13}$$

We now discuss the advanced features obtained from the above synthesizable descriptors as follows. These features are used in a calculation of a character pair distance at the text-line-formation stage mentioned in Sec 3.2.2. When the values of these features are small, a probability that the regions related to the features are characters is high. Discontinuity is a feature we introduce in this paper. Other features are similar to typical features for scene-text detection, but some have been extended with RBB to be more robust against orientation.

- **Discontinuity of Single Character**

$$d^{\mathrm{dcn}}(\boldsymbol{r}_i)= \tag{14}$$
$$\begin{cases}0 & (n^{\mathrm{cnt}}=1)\\ \sum_{m=0}^{\mathrm{N^{ori}}}\sum_{q=1}^{n^{\mathrm{cnt}}}\frac{\boldsymbol{v}^{\mathrm{min}}_{\boldsymbol{r}_i,\theta_m}[q]-\boldsymbol{v}^{\mathrm{max}}_{\boldsymbol{r}_i,\theta_m}[q-1]}{w_{\theta_m}(\boldsymbol{r}_i)} & (\mathrm{else})\end{cases},$$

  where $\boldsymbol{v}[q]$ is the $(q+1)$th element of vector $\boldsymbol{v}$, n^{cnt} is $|\boldsymbol{v}^{\mathrm{min}}_{\boldsymbol{r}_i,\theta_m}|$. Discontinuity represents the geometrical distance between CCs constituting the same compound region as shown by Fig. 4 (b). Generally, the discontinuity of correct characters is small or zero. Our method prevents an unintended compounding of non-close CCs by using the discontinuity.
- **Color Difference of Pair**

$$d^{\mathrm{clr}}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\frac{\sqrt{(d^{\mathrm{c0}})^2+(d^{\mathrm{c1}})^2+(d^{\mathrm{c2}})^2}}{255}, \tag{15}$$

  where $d^{\mathrm{c0}}=(s^{c0}(\boldsymbol{r}_i)/n^{\mathrm{ara}}(\boldsymbol{r}_i))-(s^{c0}(\boldsymbol{r}_j)/n^{\mathrm{ara}}(\boldsymbol{r}_j))$,
  $d^{\mathrm{c1}}=(s^{c1}(\boldsymbol{r}_i)/n^{\mathrm{ara}}(\boldsymbol{r}_i))-(s^{c1}(\boldsymbol{r}_j)/n^{\mathrm{ara}}(\boldsymbol{r}_j))$,
  $d^{\mathrm{c2}}=(s^{c2}(\boldsymbol{r}_i)/n^{\mathrm{ara}}(\boldsymbol{r}_i))-(s^{c2}(\boldsymbol{r}_j)/n^{\mathrm{ara}}(\boldsymbol{r}_j))$.
- **Stroke Width Difference of Pair**

$$d^{\mathrm{swd}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\frac{|swdm(\boldsymbol{r}_i)-swdm(\boldsymbol{r}_j)|}{\max(w_{\theta_m}(\boldsymbol{r}_i),h_{\theta_m}(\boldsymbol{r}_i))}, \tag{16}$$

  where $swdm(\boldsymbol{r})=s^{\mathrm{swd}}(\boldsymbol{r})/n^{\mathrm{ssp}}(\boldsymbol{r})$.
- **Vertical Overlap of Pair**

$$d^{\mathrm{ovl}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=1-\frac{y^{\mathrm{ovl}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)}{\min(h_{\theta_m}(\boldsymbol{r}_i),h_{\theta_m}(\boldsymbol{r}_j))}. \tag{17}$$

- **Height and Width Differences of Pair**

$$d^{\mathrm{h}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\frac{|h_{\theta_m}(\boldsymbol{r}_i)-h_{\theta_m}(\boldsymbol{r}_j)|}{\max(h_{\theta_m}(\boldsymbol{r}_i),h_{\theta_m}(\boldsymbol{r}_j))}, \tag{18}$$

$$d^{\mathrm{w}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\frac{|w_{\theta_m}(\boldsymbol{r}_i)-w_{\theta_m}(\boldsymbol{r}_j)|}{\max(w_{\theta_m}(\boldsymbol{r}_i),w_{\theta_m}(\boldsymbol{r}_j))}. \tag{19}$$

- **Top, Bottom, and Both Alignments of Triplet**

$$d^{\mathrm{top}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j,\boldsymbol{r}_k)= \tag{20}$$
$$\begin{cases}y^{\mathrm{diff\_t}}_{\theta_m}(\boldsymbol{r}_j,\boldsymbol{r}_k) & (\boldsymbol{r}_i=\varnothing)\\ y^{\mathrm{diff\_t}}_{\theta_m}(\boldsymbol{r}_j,\boldsymbol{r}_k)-x^{\mathrm{ctr}}_{\theta_m}(\boldsymbol{r}_j,\boldsymbol{r}_k)\frac{y^{\mathrm{diff\_t}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)}{x^{\mathrm{ctr}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)} & (\mathrm{else})\end{cases},$$

$$d^{\mathrm{btm}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j,\boldsymbol{r}_j)= \tag{21}$$
$$\begin{cases}y^{\mathrm{diff\_b}}_{\theta_m}(\boldsymbol{r}_j,\boldsymbol{r}_k) & (\boldsymbol{r}_i=\varnothing)\\ y^{\mathrm{diff\_b}}_{\theta_m}(\boldsymbol{r}_j,\boldsymbol{r}_k)-x^{\mathrm{ctr}}_{\theta_m}(\boldsymbol{r}_j,\boldsymbol{r}_k)\frac{y^{\mathrm{diff\_b}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)}{x^{\mathrm{ctr}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)} & (\mathrm{else})\end{cases},$$

$$\begin{aligned}d^{\mathrm{both}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j,\boldsymbol{r}_k)=&\\ &d^{\mathrm{top}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j,\boldsymbol{r}_k)+d^{\mathrm{btm}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j,\boldsymbol{r}_k),\end{aligned} \tag{22}$$

  where
  $x^{\mathrm{ctr}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=\frac{x^{\mathrm{max}}_{\theta_m}(\boldsymbol{r}_j)+x^{\mathrm{min}}_{\theta_m}(\boldsymbol{r}_i)-x^{\mathrm{max}}_{\theta_m}(\boldsymbol{r}_j)+x^{\mathrm{min}}_{\theta_m}(\boldsymbol{r}_i)}{2}$,
  $y^{\mathrm{diff\_t}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=y^{\mathrm{max}}_{\theta_m}(\boldsymbol{r}_j)-y^{\mathrm{max}}_{\theta_m}(\boldsymbol{r}_i)$,
  $y^{\mathrm{diff\_b}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)=y^{\mathrm{min}}_{\theta_m}(\boldsymbol{r}_i)-y^{\mathrm{min}}_{\theta_m}(\boldsymbol{r}_j)$.
- **Interval Similarity of Triplet**

$$d^{\mathrm{inter}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j,\boldsymbol{r}_k)= \tag{23}$$
$$\begin{cases}0 & (\boldsymbol{r}_k=\varnothing)\\ \frac{\max(x^{\mathrm{dis}}_{\theta_m}(\boldsymbol{r}_i,\boldsymbol{r}_j)-x^{\mathrm{dis}}_{\theta_m}(\boldsymbol{r}_j,\boldsymbol{r}_k),0)}{\max(w_{\theta_m}(\boldsymbol{r}_i),w_{\theta_m}(\boldsymbol{r}_j))} & (\mathrm{else})\end{cases}.$$

  Interval similarity represents the geometrical distance to the nearest CC that is not included in the same text. Our method promotes a compounding of close CCs by using the interval similarity.

Note that RBB models only in-place rotation but not perspective distortion. For example, values of Eq. 18 and 19 for characters belonging to the same text with perspective distortion tends to be large resulting in the characters are determined to belong to different text. The alignment features shown in Eq. 20, 21 and 22 are important for detecting such text. Because characters belonging to the same text are regularly aligned and the values of the features are low regardless of the presence or absence of perspective distortion.

3.2. Text-Detection Flow

3.2.1 Character-Candidate Detection

With our method, CCs are extracted using an MSER algorithm and classified as character candidates or non-character candidates using a sequential classifier. The classification is broken down into three stages for better computational efficiency. The first and second stages use AdaBoost[17] and the third stage uses a DNN. The CCs that are classified as characters by the preceding identifier proceed to the next stage. The features used on the first and second stages are almost the same as those in [15].

In the first stage, the classifier uses low-computational cost features and removes only clearly non-character CCs. The difference from [15] is that a minimum-rotated aspect

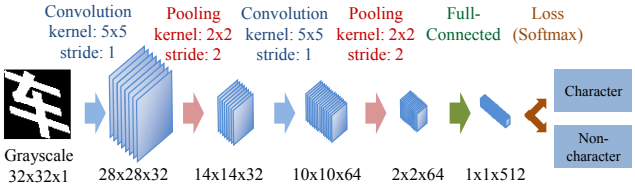

Figure 5. Structure of deep neural network

is used instead of the general aspect. The minimum-rotated aspect is defined as

$$aspect^{\min}(\boldsymbol{r}) = \min_{m\in\{0,1,\ldots,\mathrm{N^{ori}}\}} (aspect_{\theta_m}(\boldsymbol{r})). \quad (24)$$

Here, $aspect^{\min}(\boldsymbol{r})$ is robust against the rotation of $\boldsymbol{r}$ and improves the classification accuracy of rotated regions.

In the second stage, our method uses smoothness in addition to the features described in [15]. Smoothness is calculated by $s^{\mathrm{smt}}(\boldsymbol{r})/n^{\mathrm{peri}}(\boldsymbol{r})$. The second classifier outputs the class-conditional probability, and our method reduces the redundancy of detected CCs using this probability. The redundancy-reduction algorithm is similar to the linear-reduction algorithm in [27], but the difference is that the class-conditional probability is used instead of the regularized variation.

In the third stage, our method detects CCs that are characters with high probability by using the powerful discrimination performance of a DNN. Figure 5 shows the structure of the DNN used in our method. Note that all images are rotation-normalized using the angle of the minimum-aspect rotated-bounding box. After this stage, each CC has a character probability in the range of [0,1].

3.2.2 Text-Line Formation

At the text-line-formation stage, many rotated text-line candidates are detected on the basis of CCs labeled as characters with the DNN. Such a CC labeled as a character is called a base character. Our method recursively searches for neighbors of the base character using the search method to form text-line candidates. The search method is based on a character pair distance and some heuristic rules. The character pair distance represents the similarity of characters, and the distance of characters is low when the characters belong to the same text.

We now define a character pair distance for calculating the similarity of two characters. In a certain θ_m-rotated coordinate system, the distance from character region $\boldsymbol{r}_i$ to another character region $\boldsymbol{r}_j$, $d(\boldsymbol{r}_i, \boldsymbol{r}_j)$ is defined as $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_{\boldsymbol{r}_i,\boldsymbol{r}_j,\theta_m}$, where $\boldsymbol{w}$ is a parameter of weight vector. Then, $\boldsymbol{x}_{\boldsymbol{r}_i,\boldsymbol{r}_j,\theta_m}$ is a vector of ten features calculated from Eq. 14 to Eq. 23 in Sec. 3.1.2. We determine parameter $\boldsymbol{w}$ by using the distance metric learning (DML) technique [27, 25], which is a semi-supervised clustering technique and learns a distance metric that satisfies given labels. Yin *et al.* used the DML technique to automatically determine

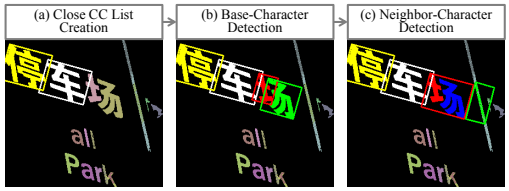

Figure 6. Flow of neighbor-character search and sample results when $\theta_m = \pi(5/12)$. (a) White region is $\boldsymbol{r}_i$, yellow one is $\boldsymbol{r}_{i-1}$, and others are in close CC list of $\boldsymbol{r}_i$. Proposed method searches neighbor character of $\boldsymbol{r}_i$. (b) Red region is initial $\boldsymbol{r}_{i+1}$ and green one is $\boldsymbol{r}_{i+2}$, which is $\boldsymbol{r}_{i+1}$'s horizontal nearest CC in CC list. $d(\boldsymbol{r}_i, \boldsymbol{r}_{i+1})$ is large because $\boldsymbol{r}_{i+1}$ and $\boldsymbol{r}_{i+2}$ are very close and $d^{\mathrm{inter}}_{\theta_m}(\boldsymbol{r}_i, \boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+2})$ is large. (c) Combination of red and blue regions is $\boldsymbol{r}_i$'s neighbor character, which is detected at end.

weights and threshold to distinguish characters from non-characters. We also used this technique to determine $\boldsymbol{w}$, but only the threshold is determined manually.

Our search method uses the pair distance to detect a neighbor characters of $\boldsymbol{r}_i$. The process is performed based on four regions including $\boldsymbol{r}_i$, the rest of which are a nearest character region of the $\boldsymbol{r}_i$, referred to as $\boldsymbol{r}_{i-1}$, and two undetermined regions, referred to as $\boldsymbol{r}_{i+1}$ and $\boldsymbol{r}_{i+2}$. $\boldsymbol{r}_{i+1}$ represents a neighbor-character candidate of $\boldsymbol{r}_i$. $\boldsymbol{r}_{i-1}$ and $\boldsymbol{r}_{i+2}$ are used to determine whether $\boldsymbol{r}_{i+1}$ should be included in the text-line candidate. At calculating $\boldsymbol{x}_{\boldsymbol{r}_i,\boldsymbol{r}_{i+1},\theta_m}$, the input of Eq. 14 is $(\boldsymbol{r}_{i+1})$, that of Eqs. 20, 21 and 22 are $(\boldsymbol{r}_{i-1}, \boldsymbol{r}_i, \boldsymbol{r}_{i+1})$, that of Eq. 23 is $(\boldsymbol{r}_i, \boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+2})$, that of other Eqs. are $(\boldsymbol{r}_i, \boldsymbol{r}_{i+1})$. Figure 6 show examples of the four regions and the flow of neighbor-character search. Our search method searches for a neighbor character of $\boldsymbol{r}_i$ by selecting and evaluating multiple-neighbor-character candidates as $\boldsymbol{r}_{i+1}$. For selecting $\boldsymbol{r}_{i+1}$ and $\boldsymbol{r}_{i+2}$, our search method constructs a close CC list of $\boldsymbol{r}_i$, as shown in Fig. 6 (a). The close CC list of $\boldsymbol{r}_i$ is a set of $\boldsymbol{r}_{cc}$ that satisfies the following conditions:

$$d^{\mathrm{clr}}_{\theta_m}(\boldsymbol{r}_{cc}, \boldsymbol{r}_i) < th^{\mathrm{clr}}_{cc}, \quad d^{\mathrm{swd}}_{\theta_m}(\boldsymbol{r}_{cc}, \boldsymbol{r}_i) < th^{\mathrm{swd}}_{cc}, \quad (25)$$

$$x^{\mathrm{dis}}_{\theta_m}(\boldsymbol{r}_{cc}, \boldsymbol{r}_i) < th^{\mathrm{hor}}_{cc}, \quad d^{\mathrm{ovl}}_{\theta_m}(\boldsymbol{r}_{cc}, \boldsymbol{r}_i) < th^{\mathrm{ver}}_{cc}. \quad (26)$$

The close CC list is used to narrow the neighbor-character search range. Each threshold, th^{clr}_{cc}, th^{swd}_{cc}, th^{hor}_{cc} and th^{ver}_{cc}, is determined as a value in which all correct samples of training data can be detected. Next, our search method selects the initial $\boldsymbol{r}_{i+1}$ from the close CC list based on heuristic rules. The initial $\boldsymbol{r}_{i+1}$ is the CC with the lowest $x^{\mathrm{dis}}_{\theta_m}(\boldsymbol{r}_i, \boldsymbol{r}_{i+1})$ that satisfies conditions $y^{\mathrm{ovl}}_{\theta_m}(\boldsymbol{r}_i, \boldsymbol{r}_{i+1}) > 0$ and $d^{\mathrm{h}}_{\theta_m}(\boldsymbol{r}_i, \boldsymbol{r}_{i+1}) > 0.5$. On the other hand, $\boldsymbol{r}_{i+2}$ is the horizontal nearest CC that satisfies $y^{\mathrm{ovl}}_{\theta_m}(\boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+2}) > 0$ in the CC list. As a result, our search method determines the four regions for calculating $\boldsymbol{x}_{\boldsymbol{r}_i,\boldsymbol{r}_{i+1},\theta_m}$.

Our search method searches for a combination with lower pair distance from the close CC list. The initial $\boldsymbol{r}_{i+1}$ becomes the initial neighbor candidate $\boldsymbol{r}^{\mathrm{cand}}_{i+1}$, and $d(\boldsymbol{r}_i, \boldsymbol{r}^{\mathrm{cand}}_{i+1})$ is calculated. Then, temporary compound regions of $\boldsymbol{r}^{\mathrm{cand}}_{i+1}$ and other CCs of the CC list are cre-

ated and pair distances are calculated. Next, the temporary compound region with the lowest distance to $\boldsymbol{r}_i$ is selected as $\boldsymbol{r}_{i+1}^{\mathrm{lowest}}$, and $\boldsymbol{r}_{i+1}^{\mathrm{cand}}$ is replaced with $\boldsymbol{r}_{i+1}^{\mathrm{lowest}}$ when $d(\boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\mathrm{cand}}) \geq d(\boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\mathrm{lowest}})$. Repeating the above operation until $d(\boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\mathrm{cand}}) < d(\boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\mathrm{lowest}})$, and the last $\boldsymbol{r}_{i+1}^{\mathrm{cand}}$ is selected as the detected neighbor character of $\boldsymbol{r}_i$, $\boldsymbol{r}^{\mathrm{nbr}}$. Finally, the method will add the detected character to the text-line candidate when $d(\boldsymbol{r}_i, \boldsymbol{r}^{\mathrm{nbr}}) < th^{\mathrm{dml}}$ and carry out neighbor-character search again. $\boldsymbol{r}_i$ and $\boldsymbol{r}_{i-1}$ are replaced with $\boldsymbol{r}^{\mathrm{nbr}}$ and $\boldsymbol{r}_i$ in the next step.

At this stage, the probability of the text-line candidate is calculated for the next text-line-elimination stage. Let t_{θ_m} be a text-line candidate whose orientation angle is θ_m and $p^{\mathrm{txt}}(t_{\theta_m})$ be the probability of t_{θ_m}. The initial value of $p^{\mathrm{txt}}(t_{\theta_m})$ is zero, and $(th^{\mathrm{dml}} - d(\boldsymbol{r}_i, \boldsymbol{r}^{\mathrm{nbr}}))$ is added to $p^{\mathrm{txt}}(t_{\theta_m})$ when $\boldsymbol{r}^{\mathrm{nbr}}$ is joined to t_{θ_m}. High $p^{\mathrm{txt}}(t_{\theta_m})$ denotes that features of character candidates constituting t_{θ_m} are very similar.

3.2.3 Text-Line Elimination

After text-line formation, too many text-line candidates remain and the majority overlap, as shown in Fig. 2 (c). Our method throttles overlapping text-line candidates into one by using the simple elimination algorithm. First, our method updates $p^{\mathrm{txt}}(t_{\theta_m})$ using the probability $p^{\mathrm{txt}}(t_{\theta_l}^{\mathrm{ovl}})$ of candidate $t_{\theta_l}^{\mathrm{ovl}}$, which overlaps t_{θ_m}. When θ_m and θ_l are the same, $p^{\mathrm{txt}}(t_{\theta_l}^{\mathrm{ovl}})$ is added to $p^{\mathrm{txt}}(t_{\theta_m})$. On the other hand, when θ_m and θ_l are different, $-p^{\mathrm{txt}}(t_{\theta_l}^{\mathrm{ovl}})$ is added to $p^{\mathrm{txt}}(t_{\theta_m})$. Next, the candidate t_{θ_m} is removed when θ_m and θ_l are different and $p^{\mathrm{txt}}(t_{\theta_m}) < p^{\mathrm{txt}}(t_{\theta_l}^{\mathrm{ovl}})$. Empirically, the number of text-line candidates with correct angles is larger than that of candidates with incorrect angles. Therefore, this elimination method works well in spite of its simplicity. Finally, the remaining text-line candidates are classified using AdaBoost with typical features [14, 23].

4. Experiments

4.1. Parameter Setting

We used the training set of the MSRA-TD500 dataset [4] to determine all parameters in our method. The MSRA-TD500 includes rotated scene text of English and Chinese. It includes a relatively large number of rotated-discontiguous characters to other datasets. For the learning of the DML technique, we labeled each character region in the training set and constructed positive and negative samples. The weight vector was determined using DML technique, and we adjusted the threshold value manually. How to learn automatically the threshold is for future work. Table 1 lists the learned parameters, which were normalized by the manually adjusted threshold value.

The training of AdaBoost and the DNN was also done using the MSRA-TD 500 training set. We determined each threshold, assuming that detecting one or more characters constituting a text is considered detecting characters constituting the text. Because our method detects a text even when only one character constituting the text is detected at the character-candidate detection stage. The labeling of all training data was done manually.

Params	w^{dcn}	w^{clr}	w^{swd}	w^{ver}	w^{h}
Leaned-value	1.30	1.02	0.395	0.603	0.415
Params	w^{w}	w^{top}	w^{btm}	w^{both}	w^{inter}
Leaned-value	0.529	0.359	0.709	0.672	0.726

Table 1. Learned weights by using MSRA-TD500 training set

In order to determine the value of $\mathrm{N}^{\mathrm{ori}}$, we evaluated recall and processing time of our method while changing $\mathrm{N}^{\mathrm{ori}}$. The recall increased with increasing $\mathrm{N}^{\mathrm{ori}}$ when $\mathrm{N}^{\mathrm{ori}} \leqq 6$ because text-line candidates with more various orientation were able to be detected when $\mathrm{N}^{\mathrm{ori}}$ is larger. However the recall did not change or decreased when $\mathrm{N}^{\mathrm{ori}} > 6$ due to an excessive detection of diagonal text candidates. Therefore we set $\mathrm{N}^{\mathrm{ori}}$ to 6. On the other hand, the processing time monotonically increased in all $\mathrm{N}^{\mathrm{ori}}$. In our experiments on MSRA-TD500 training set, the overall processing processing time in the case of $\mathrm{N}^{\mathrm{ori}} = 6$ was only 1.2 times the time in the case of $\mathrm{N}^{\mathrm{ori}} = 1$, although the number of search angles for text-line candidates is 6 times. The reason is that the descriptor calculation of compound regions had been made more efficient by using synthesizable descriptors. In our experiment, our method calculated descriptors of about 31 thousands compound regions per an image, and the total pixels of all compound regions is about 430 millions. In the case of RBB calculation, the numbers of $\min$ and $\max$ calculations were reduced from the total pixels to the number of compound regions (about $1/13,800$).

4.2. Performance Evaluation on MSRA-TD500

We evaluated our method on the MSRA-TD500 [4], and examples of detection results are given in Fig. 7 and 8. Our method could detect most of the rotated scene text that included discontiguous characters. In particular, it is usually difficult to correctly detect when some texts were dense such as in Fig. 7 (a) and (b). With our method, false positives, which overlapped with the correct text, were removed at the text-line-elimination stage, and the method succeeded in the detection. However, some text failed to be detected. The upper close texts were inadvertently connected, as shown in Fig. 8 (a). A part of the text was not detected due to light reflection, as shown in Fig. 8 (b). Two close text lines with perspective distortion were not detected, as shown in Fig. 8 (c). The cause of the detection failure is that two small characters at the right end of each text line mistakenly detected as one discontiguous character in the text-line-formation stage described in Sec. 3.2.2. The our method could not detect texts with non-uniform alignment, as shown in Fig. 8 (d) and (e), because we assumed

(a) (b) (c) (d) (e)

Figure 7. Examples of successful detections. Characters with same colored rectangles belong to same detected text line.

(a) (b) (c) (d) (e)

Figure 8. Examples of failure results. Each rectangle represents detected texts.

Method	Year	Precision	Recall	F-measure
Our method		**0.87**	0.63	**0.74**
Zhang *et al*. [28]	2016	0.83	**0.67**	**0.74**
He *et al*. [7]	2016	0.76	0.61	0.69
Yin *et al*. [27]	2015	0.81	0.63	0.71
Kang *et al*. [8]	2014	0.71	0.62	0.66
Yao *et al*. [20]	2014	0.62	0.64	0.61

Table 2. Experimental results on MSRA-TD500

that the alignment was uniform. Some repeated patterns were erroneously detected as text, as shown in Fig 7 (b) and 8 (e). In our method, text-line candidates composed of regularly aligned character-candidates have high p^{txt} and are detected as text, even when the character-candidates are not characters. The number of false-positives tends to increase when the image contains many repeated patterns, because our method searches for text at various directions. Although the detection fails due to other conditions as stated as above, our method was able to detect rotated-scene text that included discontiguous characters.

The quantitative comparisons of the proposed and existing methods are shown in Table 2 our method achieved the best score on precision and F-measure, and recall was the second best score. Our method could detect all characters constituting a text even when only one character constituting the text is detected at the character-candidate detection stage. Therefore, in the sequential CC classification stage of our method, we were able to set the threshold of the classifier to very high (the threshold of DNN was 0.95 in our implementation), and improved precision. However, the recall of our method was not high because the method was designed to detect text in a certain language such as Chinese, and only 36% of the MSRA-TD500 contains Chinese. Texts of English has a large variance of horizontal width and vertical width of characters than that of Chinese; therefore, the contribution to neighbor-character search by accurate geometrical features is smaller. We evaluated the recall for only Chinese text, which was 0.67; the same as that obtained by Zhang *et al*. [28]. The precision for only Chinese text cannot be calculated because our method detect both English and Chinese without distinction, but we consider that it is similar to the precision of Table 2. Therefore, the performance of our method is excellent for text in Chinese. Our method may also be suitable for other languages with geographical features similar to Chinese (*e.g*. Japanese).

The average detection time of the our method was about 2.28 seconds (4GHz CPU without GPU), while that of Zhang *et al*.'s method is about 2.1 seconds (2GHz CPU with GPU). We were not able to directly compare them because of different environments, but our method achieved the similar speed as their method even when we did not use GPU. The detection times of our method changed depending on an input image. The minimum time was 0.442 seconds and maximum was 20.5 seconds for the MSRA-TD500. The reason for the long detection time is that our method detected many overlapping-text-line candidates when the text was dense in the input image. Improving the detection time is for future work.

5. Conclusion

We proposed an MSER-based scene-text-detection method robust against rotated discontiguous characters. RBB accurately calculates rotated geometrical features and our neighbor-character search method with synthesizable descriptors detects discontiguous characters. Our method achieved the best score with regard to precision and F-measure on the MSRA-TD500. For Chinese text, the recall was the same as the best score of the existing methods. Therefore, we conclude that our method achieves the state-of-the-art performance with respect to precision and F-measure in detecting rotated scene-texts that include discontiguous characters. For future work, we will investigate a more suitable parameter-learning method and improvement in detection time for dense text.

References

[1] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven. Photoocr: Reading text in uncontrolled conditions. In *IEEE International Conference on Computer Vision*, pages 785–792, 2013.

[2] E. O. Boris Epshtein, Yonathan Wexler. Detecting text in natural scenes with stroke width transform. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2960–2970, 2010.

[3] X. Chen and A. L. Yuille. Detecting and reading text in natural scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 366–373, 2004.

[4] Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. Detecting texts of arbitrary orientations in natural images. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 8, pages 1083–1090. IEEE, 2012.

[5] L. Gomez and D. Karatzas. MSER-Based Real-Time Text Detection and Tracking. In *International Conference on Pattern Recognition*, pages 3110–3115. IEEE, 2014.

[6] A. R. Hanson, E. Learned-Miller, and J. J. Weinman. Scene text recognition using similarity and a lexicon with sparse belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1733–1746, 2009.

[7] T. He, W. Huang, Y. Qiao, and J. Yao. Text-attentional convolutional neural network for scene text detection. *IEEE Transactions on Image Processing*, 25(6):2529–2541, 2016.

[8] L. Kang, Y. Li, and D. Doermann. Orientation robust text line detection in natural images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4034–4041. IEEE, 2014.

[9] K. I. Kim, K. Jung, and J. H. Kim. Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1631–1639, 2003.

[10] J.-J. Lee, P.-H. Lee, S.-W. Lee, A. Yuille, and C. Koch. Adaboost for text detection in natural scene. In *International Conference on Document Analysis and Recognition*, pages 429–434, 2011.

[11] T. Lu, S. Palaiahnakote, C. L. Tan, and W. Liu. *Introduction to Video Text Detection*, pages 1–18. Springer London, London, 2014.

[12] J. Matas, O. Chum, M. Urban, and T. Pajdl. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference*, pages 36.1–36.10, 2002.

[13] R. Minetto, N. Thome, M. Cord, N. J. Leite, and J. Stolfi. Snoopertext: A text detection system for automatic indexing of urban scenes. *Computer Vision and Image Understanding*, 122:92–104, 2014.

[14] L. Neumann and J. Matas. Text localization in real-world images using efficiently pruned exhaustive search. In *International Conference on Document Analysis and Recognition*, pages 687–691. IEEE, 2011.

[15] L. Neumann and J. Matas. Real-time scene text localization and recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 38, pages 3538–3545. IEEE, 2012.

[16] L. Neumann and J. Matas. Efficient scene text localization and recognition with local character refinement. In *International Confrence on Document Analysis and Recognition*, pages 746–750, 2015.

[17] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[19] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng. End-to-end text recognition with convolutional neural networks. In *International Conference on Pattern Recognition*, pages 3304–3308, 2012.

[20] C. Yao, X. Bai, and W. Liu. A unified framework for multioriented text detection and recognition. *IEEE Transactions on Image Processing*, 23(11):4737–4749, 2014.

[21] C. Yi and Y. Tian. Localizing text in scene images by boundary clustering, stroke segmentation, and string fragment classification. *IEEE Transactions on Image Processing*, 21(9):4256–4268, 2012.

[22] C. Yi and Y. Tian. Text extraction from scene images by character appearance and structure modeling. *Computer Vision and Image Understanding*, 117(2):182–194, 2013.

[23] X. Yin, X.-C. Yin, H.-W. Hao, and K. Iqbal. Effective text localization in natural scene images with mser, geometry-based grouping and adaboost. In *International Conference on Pattern Recognition*, pages 725–728. IEEE Computer Society, 2012.

[24] X.-C. Yin, H. W. Hao, J. Sun, and S. Naoi. Robust vanishing point detection for mobilecam-based documents. In *2011 International Conference on Document Analysis and Recognition*, pages 136–140, 2011.

[25] X.-C. Yin, W. Y. Pei, J. Zhang, and H. W. Hao. Multi-orientation scene text detection with adaptive clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1930–1937, 2015.

[26] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao. Accurate and robust text detection: A step-in for text retrieval in natural scene images. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1091–1092, 2013.

[27] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao. Robust text detection in natural scene images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(5):970–983, 2014.

[28] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai. Multi-oriented text detection with fully convolutional networks. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4159–4167, 2016.

[29] S. Zhu and R. Zanibbi. A text detection system for natural scenes with convolutional feature learning and cascaded classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 625–632, 2016.